A Markovian Approach to the Induction of Regular String Distributions

Jérôme Callut and Pierre Dupont

Department of Computing Science and Engineering, INGI
Université catholique de Louvain,
Place Sainte-Barbe 2,
B-1348 Louvain-la-Neuve, Belgium
{jcal,pdupont}@info.ucl.ac.be

Abstract We propose in this paper a novel approach to the induction of the structure of Hidden Markov Models (HMMs). The notion of partially observable Markov models (POMMs) is introduced. POMMs form a particular case of HMMs where any state emits a single letter with probability one, but several states can emit the same letter. It is shown that any HMM can be represented by an equivalent POMM. The proposed induction algorithm aims at finding a POMM fitting a sample drawn from an unknown target POMM. The induced model is built to fit the dynamics of the target machine observed in the sample. A POMM is seen as a lumped process of a Markov chain and the induced POMM is constructed to best approximate the stationary distribution and the mean first passage times (MFPT) observed in the sample. The induction relies on iterative state splitting from an initial maximum likelihood model. The transition probabilities of the updated model are found by solving an optimization problem to minimize the difference between the observed MFPT and their values computed in the induced model.

Keywords: HMM topology induction, Partially observable Markov model, Mean first passage time, Lumped Markov process, State splitting algorithm.

1 Introduction

Hidden Markov Models (HMMs) are widely used in many pattern recognition areas, including applications to speech recognition [15], biological sequence modeling [6], information extraction [7,8] and optical character recognition [11], to name a few. In most cases, the model structure, also referred to as topology, is defined according to some prior knowledge of the application domain. Automatic techniques for inducing the HMM topology are interesting as the structures are sometimes hard to define *a priori* or need to be tuned after some task adaptation. The work described here presents a new approach towards this objective.

Probabilistic automata (PA) form an alternative representation class to model distributions over strings, for which several induction algorithms have been proposed. PA and HMMs actually form two families of equivalent models, according to whether or not final probabilities are included. In the former case, the models generate distributions over words of finite length, while, in the later case, distributions are defined over complete finite prefix-free sets [5].

The equivalences between PA and HMMs can be used to apply induction algorithms in either formalism to model the same classes of string distributions.

Lecture Notes in Artificial Intelligence, No. 3264, Grammatical Inference: Algorithms and Applications, pp. 77–90, October, 2004.

Nevertheless, previous works with HMMs mainly concentrated either on hand-built models (*e.g.* [7]) or heuristics to refine predefined structures [8]. More principled approaches are the Bayesian merging technique due to Stolcke [18] and the maximum likelihood state-splitting method of Ostendorf and Singer [14]. The former approach however has been applied only to small problems while the later is specific to the subclass of left-to-right HMMs modeling speech signals.

In contrast, PA induction techniques are often formulated in theoretical learning frameworks. These frameworks typically include adapted versions of the *PAC* model [16], *Identification with probability one* [1,2] or *Bayesian learning* [19]. Other approaches use error-correcting techniques [17] or statistical tests as a model fit induction bias [10]. All these approaches, while being interesting, are still somehow limited. From the theoretical viewpoint, PAC learnability is only feasible for restricted subclasses of PAs (see [5], for a review). The general PA class is identifiable with probability one [2] but this learning framework is weaker than the PAC model. In particular, it guarantees asymptotic convergence to a target model but does not bound the overall computational complexity of the learning process. From a practical viewpoint, several induction algorithms have been applied, typically to language modeling tasks [4,3,19,12]. The experiments reported in these works show that automatically induced PA hardly outperform well smoothed *discrete Markov chains* (MC), also known as *N-grams* in this context. Hence even though HMMs and PA are more powerful than simple Markov chains, it is still unclear whether these models should be considered when no strong prior knowledge can help to define their structure.

The present contribution describes a novel approach to the structural induction of HMMs. The general objective is to induce the structure and to estimate the parameters of a HMM from a sample assumed to have been drawn from an unknown target HMM. The goal however is not the identification of the target model but the induction of a model sharing with the target the main features of the distribution it generates. We restrict here our attention to features that can be deduced from the sample. These features are closely related to fundamental quantities of a Markov process, namely the *stationary distribution* and *mean first passage times*. In other words, the induced model is built to fit the dynamics of the target machine observed in the sample, not necessarily to match its structure.

We show in section 2 that any HMM can be converted into an equivalent *Partially Observable Markov Model* (POMM). Any state of a POMM emits with probability 1 a single letter, but several states can emit the same letter. Several properties of standard Markov chains are reviewed in section 3. The relation between a POMM and a lumped process in a Markov chain is detailed in section 4. This relation forms the basis of the induction algorithm presented in section 5.

2 Hidden Markov Models and Partially Observable Markov Models

We recall in this section the classical definition of a HMM and we show that any HMM can be represented by an equivalent partially observable model.

Definition 1 (HMM). *A discrete* Hidden Markov Model (HMM) *(with state emission) is a 5-tuple* $M = \langle \Sigma, Q, A, B, \iota \rangle$ *where* Σ *is an alphabet,* Q *is a set of states,* $A : Q \times Q \to [0, 1]$ *is a mapping defining the probability of each transition,* $B : Q \times \Sigma \to [0, 1]$ *is a mapping defining the emission probability of each letter on each state, and* $\iota : Q \to [0, 1]$ *is a mapping defining the initial probability of each state. The following stochasticity (or properness) constraints must be satisfied:* $\forall q \in Q, \sum_{q' \in Q} A(q, q') = 1$*;* $\forall q \in Q, \sum_{a \in \Sigma} B(q, a) = 1$*;* $\sum_{q \in Q} \iota(q) = 1$.

Figure 1 presents a HMM defined as follows:
$\Sigma = \{a, b\}$, $Q = \{1, 2\}$, $\iota(1) = 0.4; \iota(2) = 0.6$;
$A(1,1) = 0.1; A(1,2) = 0.9; A(2,1) = 0.7; A(2,2) = 0.3$;
$B(1,a) = 0.2; B(1,b) = 0.8; B(2,a) = 0.9; B(2,b) = 0.1$

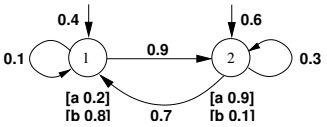

Figure 1. HMM example.

Definition 2 (HMM path). *Let* $M = \langle \Sigma, Q, A, B, \iota \rangle$ *be a HMM. A* path *in* M *is a word defined on* Q^**. For any path* ν*,* ν_i *denotes the* i*-th state of* ν*, and* $|\nu|$ *denotes the path length. For any word* $u \in \Sigma^*$ *and any path* $\nu \in Q^*$*, the probabilities* $P_M(u, \nu)$ *and* $P_M(u)$ *are defined as follows:*

$$P_M(u, \nu) = \begin{cases} \iota(\nu_1) \prod_{i=1}^{l-1} [B(\nu_i, u_i) A(\nu_i, \nu_{i+1})] B(\nu_l, u_l) \text{ if } l = |u| = |\nu| > 0, \\ 1 \text{ if } |u| = |\nu| = 0 \text{ and} \\ 0 \text{ otherwise.} \end{cases}$$

$$P_M(u) = \sum_{\nu \in Q^*} P(u, \nu).$$

$P_M(u, \nu)$ is the probability to emit word u while following path ν. $P_M(u)$ can be interpreted as the probability of observing a finite word u as part of a random walk through the model. For instance, the probability of the word ab in the HMM of Fig. 1 is given by: $P_M(ab) = P_M(ab, 11) + P_M(ab, 12) + P_M(ab, 21) + P_M(ab, 22) = 0.0064 + 0.0072 + 0.3024 + 0.0162 = 0.3322$.

Definition 3 (POMM).
A Partially Observable Markov Model (POMM) *is a HMM* $M = \langle \Sigma, Q, A, B, \iota \rangle$ *with emission probabilities satisfying:* $\forall q \in Q, \exists a \in \Sigma$ such that $B(q, a) = 1$.

In other words, any state in a POMM emits a specific letter with probability 1. Hence we can consider that POMM states only emit a single letter. This model is called *partially* observable since, in general, several distinct states can emit the same letter. As for a HMM, the observation of a word emitted during a random walk does not allow to identify the states from which each letter was emitted. However, the observations define *state subsets* from which each letter may have been emitted. Theorem 1 shows that the class of POMMs is equivalent to the class of HMMs, as any distribution generated by a HMM can be represented by a POMM.

Theorem 1 (Equivalence between HMMs and POMMs). *Let* $M = \langle \Sigma, Q, A, B, \iota \rangle$ *be a HMM, there exists an equivalent POMM* $M' = \langle \Sigma, Q', A', B', \iota' \rangle$.

Proof. Let M' be defined as follows.

- $Q' = Q \times \Sigma$,
- $B'((q,a),x) = 1$ if $x = a$, and 0 otherwise,
- $A'((q,a),(q',b)) = B(q,b)A(q,q')$,
- $\iota'((q,a)) = \sum_{q' \in Q} \iota(q')B(q',a)A(q',q)$.

It is easily shown that M' satisfies the stochasticity constraints. Let $u = u_1 \dots u_l$ be a word of Σ^* and let $\nu = ((q_1,u_1)\dots(q_l,u_l))$ be a path in M'. We have:

$$\begin{aligned} P_{M'}(u,\nu) &= \iota'((q_1,u_1)) \prod_{i=1}^{l-1} [B'((q_i,u_i),u_i)A'((q_i,u_i),(q_{i+1},u_{i+1}))] B'((q_l,u_l),u_l) \\ &= \sum_{q' \in Q} \iota(q')B(q',u_1)A(q',q_1) \prod_{i=1}^{l-1} [B(q_i,u_{i+1})A(q_i,q_{i+1})] \\ &= \sum_{q' \in Q} P_M(u,q'q_1\dots q_{l-1})A(q_{l-1},q_l) \end{aligned}$$

Summing up over all possible paths of length $l = |u|$ in M', we obtain:

$$\begin{aligned} P_{M'}(u) = \sum_{\nu \in Q'^l} P_{M'}(u,\nu) &= \sum_{\nu_1 \in Q^{l-1}} \sum_{q' \in Q} P_M(u,q'\nu_1) \sum_{q \in Q} A(q_{|\nu_1|},q) \\ &= \sum_{\nu_2 \in Q^l} P_M(u,\nu_2) = P_M(u) \end{aligned}$$

Hence, M and M' generate the same distribution. □

The proof of theorem 1 is adapted from a similar result showing the equivalence between PA without final probabilities and HMMs [5]. An immediate corollary of this theorem is the equivalence between PA and POMMs. Hence we call *regular string distribution*, any distribution generated by these models[1]. Figure 2 shows an HMM and its equivalent POMM.

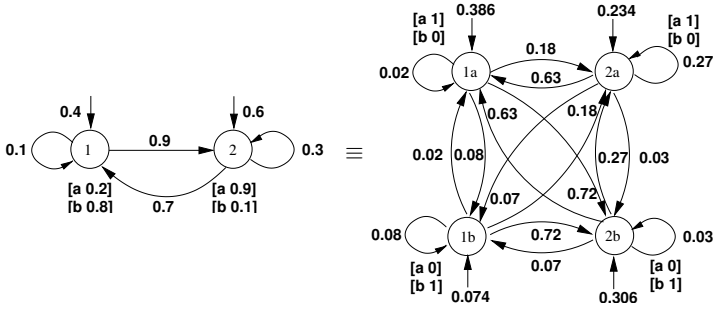

Figure 2. Transformation of a HMM into an equivalent POMM.

It should be stressed that all transition probabilities of the form $A'((q,\_),(q',b))$ are necessarily equal as the value of $A'((q,a),(q',b))$ does not depend on a in a POMM constructed in this way. A state (q,a) in this model represents the state q reached during a random walk in the original HMM after having emitted the letter a on any state.

[1] More precisely, these models generate distributions over complete finite prefix-free sets. A typical case is a distribution defined over Σ^n, for some positive integer n. See [5] for further details.

3 Markov Chains, Stationary Distribution and Mean First Passage Times

The notion of POMM introduced in section 2 is closely related to a standard Markov Chain (MC). Indeed, in the particular case where all states emit a different letter, the process of a POMM is fully observable. Moreover the Markov property is satisfied as, by definition, the probability of any transition only depends on the current state. Some fundamental properties of a Markov chain are recalled in this section. The links between a POMM and a MC are further detailed in section 4.

Definition 4 (Discrete Time Markov Chain). *A discrete time* Markov Chain (MC) *is a stochastic process* $\{X_t\}$ *where the random variable* X *takes its value at any discrete time* t *in a countable set* Q *and such that:* $P[X_{t+1} = q|X_t, X_{t-1}, \ldots, X_0] = P[X_{t+1} = q|X_t]$. *This condition states that the probability of the next outcome only depends on the last value of the process. This is known as the (first-order) Markov property. When the set* Q *is finite the process forms a* finite state MC.

Definition 5 (Finite State MC representation). *A* finite state representation of a MC *is a 3-tuple* $T = \langle Q, A, \iota \rangle$ *where* Q *is a finite set of states,* $A = Q \times Q \to [0, 1]$ *is a mapping defining the transition probability function and* $\iota : Q \to [0, 1]$ *is the initial probability of each state. The following stochasticity constraints must be satisfied:* $\sum_{q \in Q} \iota(q) = 1$*;* $\forall q \in Q, \sum_{q' \in Q} A(q, q') = 1$

In this context, the Markov property simply states that the probability of reaching the next state only depends on the current state. For a finite MC, the transition probability function can be represented as a $|Q| \times |Q|$ transition matrix. In the sequel, A both denotes this function and its matrix representation, with $A_{qq'} = A(q, q')$. Similarly, the function ι is associated with a $|Q|-$dimensional initial probability vector, with $\iota_q = \iota(q)$. We will use interchangeably MC to denote a finite Markov chain or its finite state representation. A finite MC can also be constructed from a HMM by ignoring the emission probabilities and the alphabet. We call this model the *underlying MC* of a HMM.

Definition 6 (Underlying MC of a HMM). *Given a HMM* $M = \langle \Sigma, Q, A, B, \iota \rangle$*, the* underlying Markov chain T *is the 3-tuple* $\langle Q, A, \iota \rangle$.

Definition 7 (Random walk string). *Given a MC,* $T = \langle Q, A, \iota \rangle$*, a* random walk string s *can be defined on* Q^* *as follows. A random walker is positioned on a state* q *according to the initial distribution* ι*. The random walker next moves to some state* q' *according to the probability* $A(q, q')$*. Repeating this operation* n *times results in a* n-steps random walk*. The string* s *is the sequence of states visited during this walk.*

In the present work, we focus on *regular* Markov chains. For such chains, there is a strictly positive probability to be in any state after n steps, no matter the starting state.

Definition 8 (Regular MC). *A MC with transition matrix A is* regular *if and only if for some $n \in \mathbb{N}$, the power matrix $A^{(n)}$ has no zero entries.*

In other words, the transition graph of a regular MC is *strongly connected*[2] and all states are *aperiodic*[3]. The *stationary distribution* and *mean first passage times* are fundamental quantities characterizing the dynamics of random walks in a regular MC. These quantities form the basis of the induction algorithm presented in section 5.2.

Definition 9 (Stationary distribution). *Given a regular MC, $T = \langle Q, A, \iota \rangle$, the* stationary distribution *is a $|Q|-$dimensional stochastic vector π such that $\pi A = \pi$.*

This vector is also known as the *equilibrium vector* or *steady-state vector*. A regular MC is started at equilibrium when the initial distribution ι is set to the stationary distribution π. The q-th entry of the vector π can be interpreted as an expected proportion of the time the steady-state process reaches state q.

Definition 10 (Mean First Passage Time). *Given a regular MC, $T = \langle Q, A, \iota \rangle$, the* first passage time *is a function $f = Q \times Q \to \mathbb{N}$ such that $f(q, q')$ is the number of steps before reaching state q' for the first time, leaving initially from state q.*

$$f(q, q') = \inf\{t \geq 1 \mid X_t = q' \text{and } X_0 = q\}$$

The Mean First Passage Time (MFPT) *denotes the expectation of this function. It can be represented by the MFPT matrix M, with $M_{qq'} = E[f(q, q')]$.*

For a regular MC, the MFPT values can be obtained by solving the following linear system [9]:

$$\forall q, q' \in Q, M_{qq'} = \begin{cases} 1 + \sum_{q'' \neq q'} A_{qq''} M_{q''q'} & \text{, if } q \neq q' \\ \dfrac{1}{\pi_q} & \text{, otherwise.} \end{cases}$$

The values M_{qq} are usually called *recurrence times*[4].

4 Relation between Partially Observable Markov Models and Markov Chains

Given a MC, a partition can be defined on its state set and the resulting process is said to be *lumped*.

Definition 11 (Lumped process). *Given a regular MC, $T = \langle Q, A, \iota \rangle$, let $q^{(t)}$ be the state reached at time t during a random walk in T. $\kappa = \{\kappa_1, \kappa_2, \ldots, \kappa_r\}$ denotes a partition of the set of states Q. $K_\kappa = Q \to 2^Q$ denotes a function that, given a state q, returns the block of κ, or state subset, containing q. The* lumped process *$T/\!/\kappa$ outcomes $K_\kappa(q^{(t)})$ at time t.*

[2] The chain is said to be *irreducible*.

[3] A state i is *aperiodic* if $A_{ii}^{(n)} > 0$ for all sufficiently large n.

[4] An alternative definition, $M_{qq} = 0$, is possible when it is not required to leave the initial state before reaching the destination state for the first time [13].

Consider for example the regular MC T_1 illustrated[5] in Fig. 3. A partition κ is defined on its states set, with $\kappa_1 = \{1, 3\}, \kappa_2 = \{2\}$ and $\kappa_3 = \{4\}$. The random walk 312443 in T_1 corresponds to the following observations in the lumped process $T_1/\!\!/\kappa$: $\kappa_1\kappa_1\kappa_2\kappa_3\kappa_3\kappa_1$.

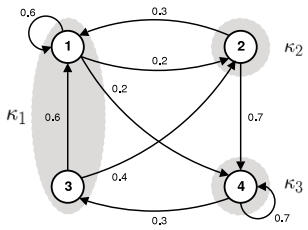

Figure 3. A regular Markov chain T_1 and the partition $\kappa = \{\{1, 3\}, \{2\}, \{4\}\}$.

While the states are fully observable during a random walk in a MC, a lumped process is associated to random walks where only state *subsets* are observed. In this sense, the lumped process makes the MC only partially observable as it is the case for a POMM. Conversely, a random walk in a POMM can be considered as a lumped process of its underlying MC with respect to an *observable partition* of its state set. Each block of the observable partition corresponds to the state(s) emitting a specific letter.

Definition 12 (Observable partition). *Given a POMM* $M = \langle \Sigma, Q, A, B, \iota \rangle$, *the* observable partition κ *is defined as follows:* $\forall q, q' \in Q, K_\kappa(q) = K_\kappa(q') \Leftrightarrow \exists a \in \Sigma, B(q, a) = B(q', a) = 1$

The underlying MC T of a POMM M has the same state set as M. Thus the observable partition κ of M is also defined for the state set of T. If each block of this partition is labeled by the associated letter, M and $T/\!\!/\kappa$ define the same string distribution.

It is important to notice that the Markov property is not necessarily satisfied for a lumped process. For example, the lumped MC in Fig. 3 satisfies $P[X_{t+2} = \kappa_2 \mid X_{t+1} = \kappa_1, X_t = \kappa_2] = 0.2$ and $P[X_{t+2} = \kappa_2 \mid X_{t+1} = \kappa_1, X_t = \kappa_3] = 0.4$, which clearly violates the *first-order* Markov property. In general, the Markov property is not satisfied when, for a fixed length history, it is impossible to decide unequivocally which state the process has reached in a given block while the next step probability differs for several states in this block. This can be the case no matter the length of the history considered. This is illustrated by the MC depicted in Fig. 4 and the partition $\kappa = \{\{1, 2\}, \{3\}\}$. Even if the complete history of the lumped process is given, there is no way to know the state reached in κ_1. Thus, the probability $P[X_t = \kappa_2 \mid X_{t-1} = \kappa_1, X_{t-2}, \ldots, X_0]$ cannot be unequivocally determined and the lumped process is not markovian for any order. Hence the definition of *lumpability*.

[5] For the sake of clarity, the initial probability of each state is not depicted. Moreover, as we are mostly interested in MC being in steady-state mode, the initial distribution is assumed to be equal to the stationary distribution deriving from the transition matrix (see Def. 9).

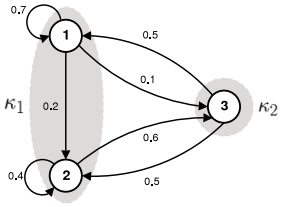

Figure 4. A non markovian lumped process.

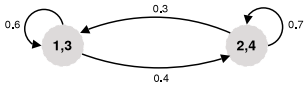

Figure 5. The MC T_1 lumped with respect to the partition $\kappa' = \{\{1,2\},\{3,4\}\}$.

Definition 13 (Lumpability). *A MC T is lumpable with respect to a partition κ if the lumped process $T/\!/\kappa$ satisfies the first-order Markov property for any initial distribution.*

When a MC T is lumpable with respect to a partition κ, the lumped process $T/\!/\kappa$ defines itself a Markov chain.

Theorem 2 (Necessary and sufficient conditions for lumpability [9]). *A MC is lumpable with respect to a partition κ if and only if for every pair of blocks κ_i and κ_j the probability $A_{ij}/\!/\kappa$ to reach some state of κ_j is equal from every state in κ_i:*

$$\forall \kappa_i, \kappa_j \in \kappa, \forall q, q' \in \kappa_i,\ A_{ij}/\!/\kappa \triangleq \sum_{q'' \in \kappa_j} A_{qq''} = \sum_{q'' \in \kappa_j} A_{q'q''}$$

The values $A_{ij}/\!/\kappa$ form the transition matrix of the lumped chain. For example, the MC T_1 given in Fig. 3 is not lumpable with respect to the partition $\kappa = \{\{1,3\},\{2\},\{4\}\}$ while it is lumpable with respect to the partition $\kappa' = \{\{1,3\},\{2,4\}\}$. The lumped chain $T_1/\!/\kappa'$ is illustrated in Fig. 5.

Even though a lumped process is not necessarily markovian, it is useful for the induction algorithm presented in section 5.2 to define the mean first passage times between the blocks of a lumped process. To do so, it is convenient to introduce some notions from absorbing Markov chains. In a MC, a state q is said to be *absorbing* if there is a probability 1 to go from q to itself. In other words, once an absorbing state has been reached in a random walk, the process will stay on this state forever. A MC for which there is a probability 1 to end up in an absorbing state is called an *absorbing MC*. In such a model, the state set can be divided into the absorbing state set Q_A and its complementary set, the transient state set Q_T. The transition submatrix between transient states is denoted A_T. A related notion is the *mean time to absorption*.

Definition 14 (Mean Time to Absorption).
Given an absorbing MC, $T = \langle \{Q_A, Q_T\}, A, \iota \rangle$, the time to absorption *is a function $g = Q_T \to \mathbb{N}$ such that $g(q)$ is the number of steps before absorption, leaving initially from a transient state q.*

$$g(q) = \inf\{t \geq 1 \mid X_t \in Q_A, X_0 = q\}$$

The Mean Time to Absorption (MTA) *denotes the expectation of this function. It can be represented by the vector* $\mathbf{z}$ *computed as* $\mathbf{z} = (I - A_T)^{-1}.\mathbf{1}$*, where* $\mathbf{1}$ *denotes a* $|Q_T|-$*dimensional vector with each component being equal to 1.*

The q-th entry of $\mathbf{z}$ represents the mean time to absorption, leaving initially from the transient state q.

Definition 15 (MFPT for a lumped process). *Given a regular MC* $T = \langle Q, A, \iota \rangle$*,* κ *a partition of* Q *and* κ_i*,* κ_j *two blocks of* κ*, an absorbing MC* T^j *is created from* T *by transforming every state of* κ_j *to be absorbing. Furthermore, let* $\mathbf{z^j}$ *be the MTA vector of* T^j*. The mean first passage time* $M_{ij}/\!\!/\kappa$ *from* κ_i *to* κ_j *in the lumped process* $T/\!\!/\kappa$ *is defined as follows:*

$$M_{ij}/\!\!/\kappa = \begin{cases} \dfrac{1}{\pi_{\kappa_i}} & \text{if } \kappa_i = \kappa_j \\ \displaystyle\sum_{q \in \kappa_i} \frac{\pi_q}{\pi_{\kappa_i}} z_q^j & \text{otherwise} \end{cases}$$

where π_q *is the stationary distribution of state* q *in* T *and* $\pi_{\kappa_i} = \sum_{q \in \kappa_i} \pi_q$ *is the stationary distribution of the block* κ_i *in the lumped process* $T/\!\!/\kappa$*.*

In a lumped process, states subsets are observed instead of the original states of the Markov chain. A related, but possibly different, process is obtained when the states of the original MC are *merged* to form a *quotient Markov chain*.

Definition 16 (Quotient MC). *Given a MC* $T = \langle Q, A, \iota \rangle$ *and a partition* $\kappa = \{\kappa_1, \kappa_2, \ldots, \kappa_r\}$ *on* Q*, the quotient* T/κ *is a* r*-states MC with transition matrix* A/κ *and initial vector* I/κ *defined as follows:*

$$A_{ij}/\kappa = \sum_{q \in \kappa_i} \sum_{q' \in \kappa_j} \frac{\pi_q}{\pi_{\kappa_i}} A_{qq'},\ I_i/\kappa = \sum_{q \in \kappa_i} \iota(q)$$

where π *is the stationary distribution of* T *and* $\pi_{\kappa_i} = \sum_{q \in \kappa_i} \pi_q$*.*

Note that for any regular MC T, the quotient T/κ has always the Markov property while, as mentioned before, this is not necessarily the case for the lumped process $T/\!\!/\kappa$. The following theorem specifies under which condition the distributions generated by T/κ and $T/\!\!/\kappa$ are identical.

Theorem 3. *If a MC* T *is lumpable with respect to a partition* κ *then* T/κ *and* $T/\!\!/\kappa$ *generate the same distribution in steady-state.*

Proof. When T is lumpable with respect to κ, the transition probabilities between any pair of blocks κ_i, κ_j are the same in both models:

$$A_{i,j}/\kappa = \textstyle\sum_{q \in \kappa_i} \frac{\pi_q}{\pi_{\kappa_i}} \sum_{q' \in \kappa_j} A_{qq'} = A_{ij}/\!\!/\kappa \sum_{q \in \kappa_i} \frac{\pi_q}{\pi_{\kappa_i}} = A_{ij}/\!\!/\kappa$$

□

5 A Markovian Approach to the Induction of Regular Distributions

As explained in section 4, a random walk in a POMM can be seen as a lumped process of its underlying MC lumped with respect to the observable partition. We present now an induction algorithm making use of this relation. Given a data sample, assumed to have been drawn from a target POMM TP, our induction algorithm estimates a model EP fitting the dynamics of the MC related to TP. The estimation relies on the stationary distribution and the mean first passage times which can be derived from the sample. In the present work, we focus on distributions that can be represented by POMMs without final probabilities and with regular underlying MC. Since the target process TP never stops, the sample is assumed to have been observed in steady-state. Furthermore, as the transition graph of TP is strongly connected, it is not restrictive to assume that the data is a unique finite string s resulting from a random walk through TP observed during a finite time[6]. Under these assumptions, all transitions of the target POMM and all letters of its alphabet will tend to be observed in the sample. Such a sample can be called *structurally complete*. The sample estimates are detailed in section 5.1 and an algorithm for POMMs induction is proposed in section 5.2.

5.1 Sample Estimates

As the target process TP can be considered as a lumped process, each letter of the sample s is associated to a unique state subset of the observable partition κ. All estimates introduced here are related to the state subsets of the target lumped process. First, we introduce the *stationary maximum likelihood model*. This model is the starting point of the induction algorithm presented in section 5.2.

Definition 17 (Stationary maximum likelihood MC). *Given a string s on an alphabet Σ, the stationary maximum likelihood MC $ML = \langle Q, \hat{A}, \hat{\iota} \rangle$ is defined as follows: $Q = \Sigma$; $\forall a, b \in Q, \hat{A}_{ab} = \frac{count(a,b)}{count(a)}$; $\forall a \in Q, \hat{\iota}_a = \hat{\pi}_a$; where $count(a, b)$ is the number of times the letter a is immediately followed by the letter b in s, $count(a) = \sum_{b \in \Sigma} count(a, b)$ and $\hat{\pi}$ is the stationary vector computed from $\hat{A}$ (see Def. 9).*

The ML model is a maximum likelihood estimate of the quotient MC TP/κ, where κ is the observable partition. Furthermore the stationary distribution of TP/κ fits the letter distribution observed in the sample. The letter distribution is however not sufficient to reproduce the dynamics of the target machine. For instance, if the letters of s were alphabetical sorted, the stationary distribution of the ML model would be unchanged. In order to better fit the target dynamics, the induced model is further required to comply with the MFPT between the blocks of $TP/\!\!/\kappa$, that is between the letters observed in the sample.

Definition 18 (MFPT matrix estimate). *Given a string* s *defined on an alphabet Σ, $\hat{M}$ is a $|\Sigma| \times |\Sigma|$ matrix where $\hat{M}_{ab}$ is the average number of symbols after an occurrence of* a *in* s *to observe the first occurrence of* b.

[6] The statistics described in section 5.1 could equivalently be computed from repeated finite samples observed in steady-state.

5.2 Induction Algorithm

Given a target POMM TP and a random walk string s built from TP, the objective of our induction algorithm is to construct a model fitting the stationary distribution and the MFPT estimated from the sample. This algorithm starts from the stationary maximum likelihood model ML, which complies with the stationary distribution. Iterative state splitting in the current model allows to increase the fit to the MFPT, while preserving the stationary distribution. The induction algorithm is sketched hereafter.

Algorithm MarkovianStateSplit
Input: A string s resulting from a target POMM TP
A precision parameter ϵ
Output: A POMM EP

$EP \leftarrow$ `estimateML`(s); *// Build the ML model (see Def. 17)*
$\hat{M} \leftarrow$ `sampleMFPT`(s); *// MFPT between the blocks of $TP/\!/\kappa$ (Def. 18)*
$M/\!/\kappa \leftarrow$ `blockMFPT`(EP); *// MFPT between the blocks of EP (Def. 15)*

// Iterate till the MFPT of the current model are close enough to those estimated from s
while $\sum_{i,j=1}^{|\Sigma|}(\hat{M}_{ij} - M_{ij}/\!/\kappa)^2 \geq \epsilon$ **do**
  $q \leftarrow$ `selectStateToSplit`$(EP, \hat{M}, M/\!/\kappa)$;
  $EP \leftarrow$ `splitState`$(EP, q, \hat{M}, M/\!/\kappa)$; *// Update the current model*
  $M/\!/\kappa \leftarrow$ `blockMFPT`(EP); *// Recompute MFPT between the blocks of EP*
return EP

At each iteration, a state q is selected by the function `selectStateToSplit` in an arbitrary order. During the call to `splitState`, the state q is split into two new states q_1 and q_2 as depicted in Fig. 6. The *input states* $i_1, \ldots, i_k$ and *output states* $o_1, \ldots, o_l$ are those directly connected to q in the current model in which all transitions probabilities A are known. Input and output states are not necessarily disjoint.

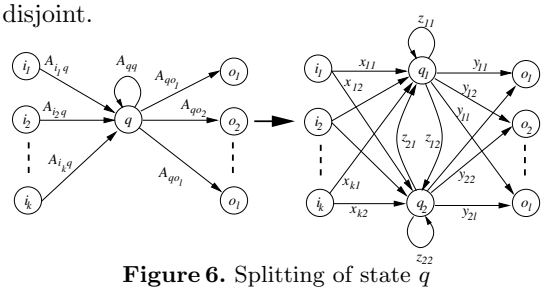

Figure 6. Splitting of state q

The topology after splitting provides additional degrees of freedom in the transition probabilities. The new transitions probabilities x, y, z form the variables of an optimization problem, which can be represented by the matrices X $(k \times 2), Y$ $(2 \times l)$ and Z (2×2). The objective function to be *minimized* is $W(X, Y, Z) = \sum_{i,j=1}^{|\Sigma|}(\hat{M}_{ij} - M_{ij}/\!/\kappa)^2$. In other words, the goal is to find values for X, Y and Z such that the MFPT of the new model are as close as possible to those estimated from s. After the splitting of state q, the `blockMFPT` function

recomputes the MFPT between the blocks of EP. The algorithm is iterated until the squared difference of the MFPT between $TP/\!\!/\kappa$ and EP fall below the precision threshold ϵ.

Stochastic constraints have to be satisfied in order to keep a proper POMM. Moreover we require the stationary distribution to be preserved for any state $q' \neq q$ and $\pi_{q_1} = \pi_{q_2} = \frac{\pi_q}{2}$. All these constraints can easily be formulated on the problem variables:

$$\forall j = 1, \ldots, k : \; x_{j1} \geq 0, x_{j2} \geq 0, \; x_{j1} + x_{j2} = A_{i_j q};$$
$$\forall j = 1, \ldots, l : \; y_{1j} \geq 0, y_{2j} \geq 0, \; y_{1j} + y_{2j} = 2A_{qo_j};$$
$$z_{11}, z_{12}, z_{21}, z_{22} \geq 0, \; z_{11} + z_{12} + z_{21} + z_{22} = 2A_{qq};$$
$$z_{11} + z_{12} + \sum_{j=1}^{l} y_{1j} = 1, \; z_{21} + z_{22} + \sum_{j=1}^{l} y_{2j} = 1.$$

6 Conclusion and Future Work

We propose in this paper a novel approach to the induction of HMMs. Firstly, the notion of partially observable Markov models (POMMs) is introduced. They form a particular case of HMMs where any state emits a single letter with probability one, but several states can emit the same letter. It is shown that any HMM can be represented by an equivalent POMM. Our induction algorithm aims at finding a POMM fitting a sample drawn from an unknown target POMM. The induced model is built to fit the dynamics of the target machine observed in the sample, not necessarily to match its structure. To do so, a POMM is seen as a lumped process of a Markov chain and the induced POMM is built to fit the stationary distribution and the mean first passage times (MFPT) observed in the sample.

Our ongoing work includes several issues. The `selectStateToSplit` function defines the order in which states are selected for splitting in our induction algorithm. Among all candidate states for splitting, the one providing the largest decrease of the objective function after the split could be considered in the first place. A simple implementation would compute the values of the objective function for all candidate states and would then select the best candidate. More efficient ways for computing this optimal state are under study.

A general solver could be used to solve the optimization problem at each iteration of the MarkovianStateSplit algorithm. An efficient implementation of this optimization procedure is under development.

A systematic experimental study of the proposed approach is our very next task. We will focus in particular on practical comparisons with standard probabilistic automata induction algorithms and EM estimation of HMMs using greedy approaches to refine predefined structures. Other perspectives include a formal study of the convergence of this approach as a function of the precision parameter ϵ and extensions to models for which the underlying Markov chain is no longer assumed to be regular.

References

1. R. Carrasco and J. Oncina. Learning deterministic regular gramars from stochastic samples in polynomial time. *Theoretical Informatics and Applications*, 33(1):1–19, 1999.

2. F. Denis and Y. Esposito. Learning classes of probabilistic automata. In *Proc. of 17th Annual Conference on Learning Theory (COLT)*, number 3120 in Lecture Notes in Computer Science, pages 124–139, Banff, Canada, 2004. Springer Verlag.
3. P. Dupont and J.C. Amengual. Smoothing probabilistic automata: an error-correcting approach. In A. Oliveira, editor, *Grammatical Inference: Algorithms and Applications*, number 1891 in Lecture Notes in Artificial Intelligence, pages 51–64, Lisbon, Portugal, 2000. Springer Verlag.
4. P. Dupont and L. Chase. Using symbol clustering to improve probabilistic automaton inference. In *Grammatical Inference, ICGI'98*, number 1433 in Lecture Notes in Artificial Intelligence, pages 232–243, Ames, Iowa, 1998. Springer Verlag.
5. P. Dupont, F. Denis, and Y. Esposito. Links between Probabilistic Automata and Hidden Markov Models: probability distributions, learning models and induction algorithms. *Pattern Recognition*, 2004. to appear in Special Issue on Grammatical Inference Techniques & Applications.
6. R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis*. Cambridge University Press, 1998.
7. D. Freitag and A. McCallum. Information extraction with hmms and shrinkage. In *Proc. of the AAAI-99 Workshop on Machine Learning for Information Extraction*, 1999.
8. D. Freitag and A. McCallum. Information extraction with HMM structures learned by stochastic optimization. In *Proc. of the Seventeenth National Conference on Artificial Intelligence, AAAI*, pages 584–589, 2000.
9. J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. Springer-Verlag, 1983.
10. C. Kermorvant and P. Dupont. Stochastic grammatical inference with multinomial tests. In P. Adriaans, H. Fernau, and M. van Zaanen, editors, *Proceedings of the 6th International Colloquium on Grammatical Inference: Algorithms and Applications*, number 2484 in Lecture Notes in Artificial Intelligence, pages 149–160, Amsterdam, the Netherlands, September 2002. Springer Verlag.
11. E. Levin and R. Pieraccini. Planar hidden markov modeling: from speech to optical character recognition. In C.L. Giles, S.J. Hanton, and J.D. Cowan, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 731–738. Morgan Kauffman, 1993.
12. D. Llorens, J.-M. Vilar, and F. Casacuberta. Finite state language models smoothed using n-grams. *International Journal of Pattern Recognition and Artificial Intelligence*, 16(3):275–289, 2002.
13. J. R. Norris. *Markov Chains*. Cambridge University Press, United Kingdom, 1997.
14. M. Ostendorf and H. Singer. Hmm topology design using maximum likelihood successive state splitting. *Computer Speech and Language*, 11:17–41, 1997.
15. L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, 1993.
16. D. Ron, Y. Singer, and N. Tishby. Learning probabilistic automata with variable memory length. In *Proceedings of the Seventh Annual Conference on Computational Learning Theory*, New Brunswick, NJ, 1994. ACM Press.
17. H. Rulot and E. Vidal. An efficient algorithm for the inference of circuit-free automata. In G. Ferratè, T. Pavlidis, A. Sanfeliu, and H. Bunke, editors, *Advances in Structural and Syntactic Pattern Recognition*, pages 173–184. NATO ASI, Springer-Verlag, 1988.
18. A. Stolcke. *Bayesian Learning of Probabilistic Language Models*. Ph. D. dissertation, University of California, 1994.
19. F. Thollard, P. Dupont, and C. de la Higuera. Probabilistic DFA Inference using Kullback-Leibler Divergence and Minimality. In *Seventeenth International Conference on Machine Learning*, pages 975–982. Morgan Kauffman, 2000.